IE 529 Final Project

Group Lasso Study Report

Team member: Jinghong Li, Han Wang, Junyi Xie

1. Introduction

The least absolute square shrinkage and selection operator (LASSO) is a regression analysis method that focuses on performing variable selection and regularization in order to enhance the prediction accuracy and interpretability of the resulting statistical model. But LASSO would reach certain limits when solving high dimensional data and potential data groups with distinctive parameters, and more obstacles emerges when dealing with high dimensional data with sparse structures: regular linear regression analysis like ridge and lasso may not generate meaningful results for interpretation when number of parameters may be much larger than sample sizes. Moreover, performance of the lasso method depends on individual dummy variables instead of whole factors, and how this dummy variable was encoded will affect results. Hence, we can't reply on the regular lasso method to examine data sets containing both continuous and categorical variables. Therefore, the idea and algorithm of the group lasso was introduced and extended. The intention of group LASSO was to do variable selection on predefined linear regression models. But under the groupwise orthogonal reparameterizations, the group lasso is shown to have the attractive property of being invariant.

The appealing part of group lasso resides in multiple aspects. First, the extension from lasso makes a linear regression method that can perform meaningful logistic regression analysis. Such proposed algorithms not only can solve generalized linear models, for example, with convex optimization problems, but also with logistic regression. Second, it can construct a hierarchical estimate using a two-stage procedure, making group lasso more suitable for analysis of DNA splice sites. Where to slice intron and exons for genes to express themselves always troubles scientists, but group lasso may provide some insights on this issue. Third, the suitability for group lasso can be a preferred choice when dealing higher order interaction terms or basis expansions for logistic additive models where the groups correspond to the basis functions for individual continuous covariates. Group lasso has more potential with high dimension data analysis applications.

2. Technical background

1) Lasso

The least absolute square shrinkage and selection operator (LASSO): a linear regression method, uses a penalty term to shrink data value towards a central point, like the mean. The lasso procedure encourages simple, sparse model (model with fewer parameters) and suitable for variable selection/parameter elimination or showing high levels of multicollinearity.

2) Logistic Group Lasso:

One key assumed technical "fact" is that the data points in p-dimension has G predictors/group of variables. For independent and identically distributed observations (Xi, Yi), I – 1, …, n, of p-dimensional vector Xi belongs to Rp of G predictors and a binary response variable Yi contains only {0,1}. Both categorical and continuous predictors are allowed. We denote by dfg the degree

of freedom of the gth predictor. For example, the main effect of a factor with four levels has df = 3 whereas a continuous predictor involves df =1 only.

3. Summary of main results

In this paper, two efficient algorithms for solving logistic regression problems using group lasso are proposed. Both algorithms are based on recent developments of theory including Block co-ordinate descent and Block co-ordinate gradient descent.

1) Set up the logistic group lasso problem.

In the setup part, several definitions are made 1) degree of freedom of the g_{th} group of predictors df_g, 2) convex function $S_\lambda(\beta)$ including log-likelihood function and 'groupwise' l_2-norm, 3) 'groupwise' l_2-norm $\lambda \sum_{g=1}^{G} s(df_g)||\beta_g||_2$ that has a similar effect as l_1-norm but is an intermediate between lasso and ridge penalty function.

2) Logistic group lasso algorithm using Block co-ordinate descent minimization

The critical point of the convergence is based on the separable structure of the non-differentiable part in $S_\lambda(\cdot)$. The step 2 and 3 in the proposed algorithm perform the groupwise minimization of $S_\lambda(\cdot)$ and the corresponding minima can be found. In general, the algorithm will perform a groupwise cycle through all given groups. At each iteration, only the current group of parameters will change while others will keep fixed. The goal is to minimize $S_\lambda(\cdot)$. The disadvantage of this algorithm is that it is not suitable for the large-scale problem due to blockwise minimization.

3) Logistic group lasso algorithm using Block co-ordinate gradient descent minimization

The main difference between BCGD and BCD algorithms is the combination of quadratic approximation of log-likelihood with line search using the Armijo rule. The Hessian of $l(\cdot)$ is replaced by the matrix $H^{(t)}$ using second-order Taylor series expansion at $\hat{\beta}^{(t)}$. The goal is to minimize $M_\lambda^{(t)}(\cdot)$ with respect to a penalized group. For high dimension situation, the remaining blocks of sets can be visited every 10 iterations to update the current set. It turns out that it is useful for saving computational time.

Note:*

It is worth mentioning that in both algorithms, we do not penalize intercept β_0.

4. Application example

1) Implement the algorithm and discuss

Functions used--

Nhessian -- a function with arguments x, mu and weights implementing the *negative* hessian of the log-likelihood function.

Ngradient -- a function with arguments x, y, mu and weights implementing the *negative* gradient of the log-likelihood function.

Nloglik -- a function with arguments y, mu and weights implementing the *negative* log-likelihood function.

Zapsmall – a function determines a digits argument for calling round such that values close to zero (compared with the maximal absolute value) are 'zapped', i.e., replaced by 0.

Pseudocode--

Set $\beta \in R^{p+1}$ be an initial parameter vector

While some convergence criterion is met:

For g=0, …, G:

Set $H_{gg} = h_g(\beta) I_{df_g}$

Set $d = argmin_{d|d_k=0, k\neq g} M_\lambda(d)$

If $d \neq 0$:

do line search for α

$\beta = \beta + \alpha d$

Step1:

Using a second-order Taylor series expansion at $\hat{\beta}^t$ and replacing the Hessian of the log-likelihood function $l(\cdot)$ of the log-likelihood function $l(\cdot)$ by a suitable matrix $H(t)$.

And for the unpenalized intercept this is not necessary, the solution can be directly computed by:

$$d_0^{(t)} = -\frac{1}{h_0^{(t)}} \nabla l\left(\hat{\beta}^t\right)_0$$

```
if(update.hess == "lambda" & pos %% update.every == 0 | pos == 1){
    ## Non-penalized groups
    if(any.notpen){
      for(j in 1:nrnotpen){ ## changed
        Xj <- x.notpen[[j]]
        nH.notpen[j] <- min(max(nhessian(Xj, mu, weights, ...), lower), upper)
      }
    }
    ## Penalized groups
    for(j in 1:nrpen){
      ind <- ipen.which[[j]]
      Xj  <- x.pen[[j]]
      diagH <- numeric(length(ind))
```

```
      for(i in 1:length(ind)){
        diagH[i] <- nhessian(Xj[, i, drop = FALSE], mu, weights, ...)
      }
      nH.pen[j] <- min(max(diagH, lower), upper)
    }
  }
```

Step2:

```
if(any.notpen){
      ## Optimize the *non-penalized* parameters
      for(j in 1:nrnotpen){
        ind <- inotpen.which[j]
        Xj  <- x.notpen[[j]]

        ## Gradient of the negative log-likelihood function
        ngrad <- c(ngradient(Xj, y, mu, weights, ...))

        ## Update the Hessian if necessary
        if(update.hess == "always"){
          nH <- min(max(nhessian(Xj, mu, weights, ...), lower), upper)
        }else{
          nH <- nH.notpen[j]
        }

        ## Calculate the search direction
        d <- -(1 / nH) * ngrad
        ## Set to 0 if the value is very small compared to the current
        ## coefficient estimate
```

If $d(t) \neq 0$, an inexact line search using the Armijo rule must be performed: let $\alpha^{(t)}$ be the largest value in $\{\alpha_0 \delta^l\}_{l \geq 0}$ such that

$$S_\lambda\left(\hat{\beta}^{(t)} + \alpha^{(t)} d^{(t)}\right) - S_\lambda\left(\widehat{\beta^{(t)}}\right) \leq \alpha^{(t)} \sigma \Delta^{(t)}$$

```
## If d! = 0, we have to do a line search
        if(d != 0){
          scale <- min(start.notpen[j] / beta, 1) ##1
          coef.test      <- coef
          coef.test[ind] <- coef[ind] + scale * d

          Xjd       <- Xj * d
          eta.test  <- eta + Xjd * scale
```

```
if(line.search){
  qh    <- sum(ngrad * d)

  fn.val0     <- nloglik(y, eta, weights, ...)
  fn.val.test <- nloglik(y, eta.test, weights, ...)

  qh <- zapsmall(c(qh, fn.val0), digits = 16)[1]

  ## Armijo line search. Stop if scale gets too small (10^-30).
  while(fn.val.test > fn.val0 + sigma * scale * qh & scale > 10^-30){
    ##cat("Doing line search (nonpen)\n")
    scale         <- scale * beta
    coef.test[ind] <- coef[ind] + scale * d
    eta.test      <- eta + Xjd * scale
    fn.val.test    <- nloglik(y, eta.test, weights, ...)
  }
} ## end if(line.search)
if(scale <= 10^-30){ ## Do nothing in that case
  start.notpen[j] <- 1
}else{ ## Update the information
  coef <- coef.test
  eta  <- eta.test
  mu   <- invlink(eta)
  start.notpen[j] <- scale
}

  ## Save the scaling factor for the next iteration
} ## end if(abs(d) > sqrt(.Machine$double.eps))
```

2) Describe application and data used, including how data is collected and/or created.

The prediction of clients' consuming behavior plays an important role in business analysis, most online shopping websites collect billions of data for pursing a better prediction result, and then optimize clients' shopping experience in their platforms.

Our website visit records dataset consists of a train set of 3561 true (encoded as Converted = 1) and 5679 false (encoded as Converted = 0) records. The target variable for this dataset is "Converted" which tells us if a past lead was converted or not, wherein 1 means it was converted and 0 means it wasn't converted.

We split the original dataset into training, validation, and testing datasets by the ratio 0.7:0.15:0.15. All records are chosen randomly without replacement such that two datasets are disjoint.

The names of columns in the dataset show below:

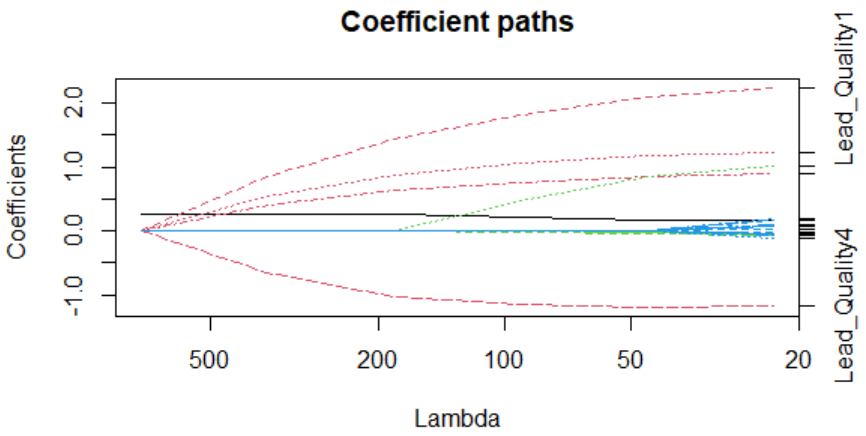
```
Index(['Prospect ID', 'Lead Number', 'Lead Origin', 'Lead Source',
       'Do Not Email', 'Do Not Call', 'Converted', 'TotalVisits',
       'Total Time Spent on Website', 'Page Views Per Visit', 'Last Activity',
       'Country', 'Specialization', 'How did you hear about X Education',
       'What is your current occupation',
       'What matters most to you in choosing a course', 'Search', 'Magazine',
       'Newspaper Article', 'X Education Forums', 'Newspaper',
       'Digital Advertisement', 'Through Recommendations',
       'Receive More Updates About Our Courses', 'Tags', 'Lead Quality',
       'Update me on Supply Chain Content', 'Get updates on DM Content',
       'Lead Profile', 'City', 'Asymmetrique Activity Index',
       'Asymmetrique Profile Index', 'Asymmetrique Activity Score',
       'Asymmetrique Profile Score',
       'I agree to pay the amount through cheque',
       'A free copy of Mastering The Interview', 'Last Notable Activity'],
      dtype='object')
```

To apply on our function and analysis, we need to do some preprocessing work before:

- Remove all columns contain less than 3 levels
- Replace all null values with NA
- Remove variables that we think won't be necessary for this analysis (e.g., City, Country)

3) Discuss the statistics of the application results and data

The solution path of group lasso object. The x-axis is the penalty parameter lambda, the y-axis is coefficients the coefficient groups of Lead_Quality and Lead_Origin.



The correlation coefficient ρ_τ corresponding to a threshold τ is defined as the Pearson correlation between the binary random variable of the true class membership and the binary random variable of the predicted class membership. The maximal correlation coefficient $\rho_{max} =$

$max\{\rho_\tau|\tau \in (0,1)\}$ can be used as a goodness-of-fit statistic on the test set of our website visit records dataset. The group lasso estimator model with respect to the log-likelihood score, has a corresponding value of ρ_{max} = 0.5375. The precision and recall of the test dataset are 0.6452, 0.6730, which are slightly smaller than the classical logistic regression 0.7829, 0.7646. We speculate that the performance is because our dataset is not enough high dimensional, and the features equal or less than 2 must be removed to apply on the model.

4) State what coding language is used, and if relevant why.

We use R as the coding language, because the paper is published in 2008, most of works relating to group lasso are realize by R language, they help us have a complete idea of how to implement quicker.

5) Provide a rough complexity analysis for the algorithm.

The computational cost of gradient descent depends on the number of iterations it takes to converge, the complexity of gradient descent is O(kn^2). Within each iteration, we must update for each group, when it do two-order taylor series expansion, the hessian matrix $H_{gg} = h_g(\beta)I_{df_g}$. For making sure convergence, the argument of $h_g$is $h_g^{(t)} = -max\left[diag - \nabla^2 l\left(\widehat{\beta^{(t)}}\right)_t, c^*\right]$, c^* is a constant as a bound to make sure convergent. Then check the KKT conditions to make sure no new groups should enter. The overall calculation appears to be dominated by the QR decomposition and gradient descent, the time complexity should be O(kn^2).

5. Impact and connections to course

In the lecture, we are introduced to both logistic regression and lasso. Several methods to solve logistic regression problems were discussed such as the steepest descent method and Newton-Raphson (second-order) method. As for lasso, the application in linear regression was discussed and the feature selection property was introduced.

The paper generalized the lasso method to the logistic regression problem. The content is somewhat too profound to understand, especially the algorithm parts. However, the setup part, i.e., set up the logistic group lasso problem, is worth studying since it is a great extension of the lasso method and the 'groupwise' l_2-norm is helpful to understand the construct of the logistic group lasso problem and the ridge regression we discussed in the lecture is also useful in understanding this content.

Reference

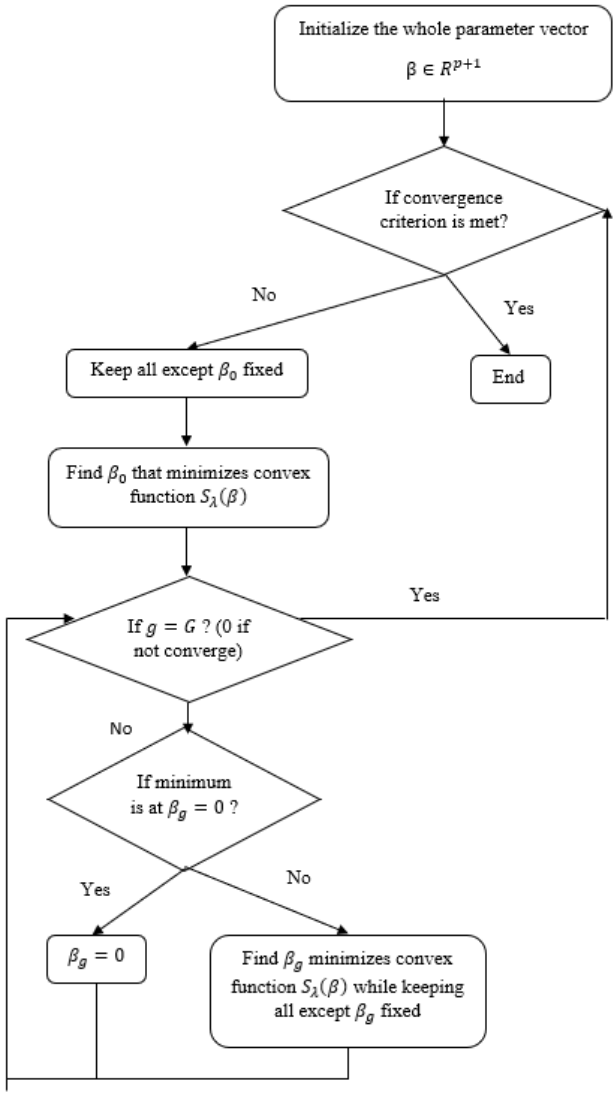
[1] Qin, Z., Scheinberg, K., & Goldfarb, D. (2013). Efficient block-coordinate descent algorithms for the group Lasso. Mathematical Programming Computation, 5(2), 143–169.

[2] Yuval Nardi, Alessandro Rinaldo "On the asymptotic properties of the group lasso estimator for linear models," Electronic Journal of Statistics, Electron. J. Statist. 2(none), 605-633, (2008)

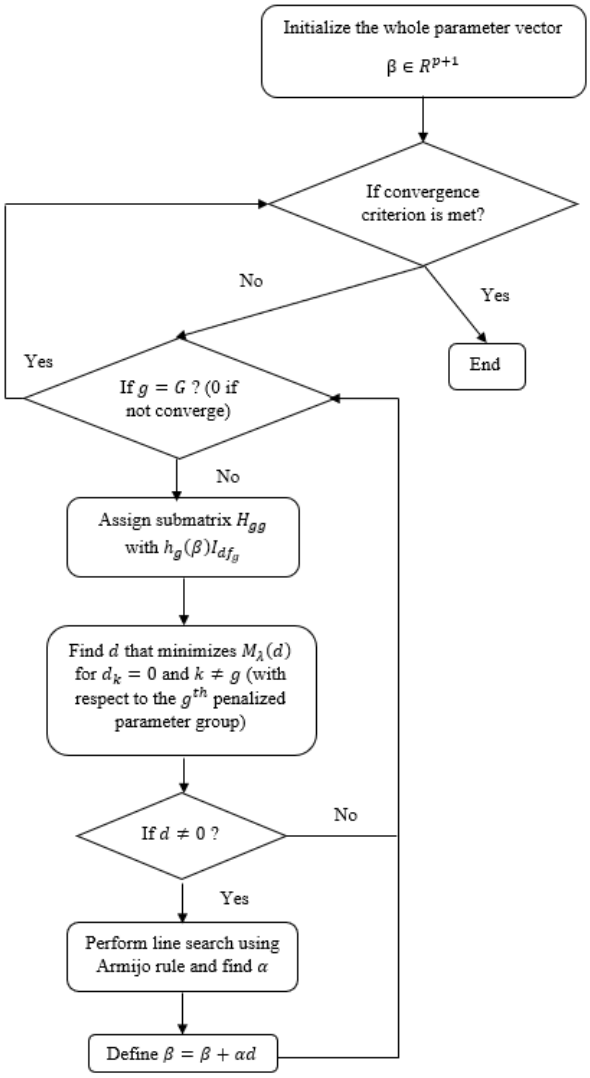
[3] Meier, L., Van De Geer, S., & Bühlmann, P. (2008). The group lasso for logistic regression. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 70(1), 53–71.

Appendix (flowchart of the main algorithms)

1. Logistic group lasso algorithm using Block co-ordinate descent minimization



2. Logistic group lasso algorithm using Block co-ordinate descent minimization


Initialize the whole parameter vector
$\beta \in R^{p+1}$
If convergence criterion is met?
No
Yes
End
Yes
If $g = G$? (0 if not converge)
No
Assign submatrix H_{gg} with $h_g(\beta)I_{df_g}$
Find d that minimizes $M_\lambda(d)$ for $d_k = 0$ and $k \neq g$ (with respect to the g^{th} penalized parameter group)
If $d \neq 0$?
No
Yes
Perform line search using Armijo rule and find α
Define $\beta = \beta + \alpha d$